Recent Progress in Convergence Improvements for OVERFLOW

Joseph M. Derlaga and Pieter G. Buning
NASA LaRC
14th Overset Grid Symposium
October 2nd, 2018
Outline

• Motivation/Background
• Improvements
• Test Cases
• Future Work
• Questions
Motivation

• As grid sizes grow, and numerical problems become more stiff, the more that can be done to improve the linear solver the better

• Looking at typical convergence plots, and based on discussions of typical CFL numbers, the current linear solvers leave a lot to be desired

• What can be done to improve meanflow convergence without fundamentally changing how OVERFLOW’s source code is structured?
Background

• OVERFLOW originally started out using finite difference discretizations, but evolved to include finite volume style discretizations as well
  • Central schemes and Steger-Warming FVS FD: 1980s
  • MUSCL FV: 1990s
  • WENO FD/FV: 2000s

• However, all the linearizations that have driven the linear solvers have remained fixed in the original finite difference framework
Background

• OVERFLOW has a mixture of spatially factored and unfactored linear solver schemes available
  • Two factor F3D scheme
  • Two factor LU-SGS
  • Three factor Beam-Warming block tridiagonal with either central difference or Steger-Warming flux linearizations
  • Three factor scalar pentadiagonal (aka, Pulliam-Chaussee or DADI)
  • Three factor Diagonalized Diagonally Dominant ADI (D3ADI)
  • Unfactored SSOR
Background

• OVERFLOW has a mixture of spatially factored and unfactored linear solver schemes available
  • Two factor F3D scheme
  • Two factor LU-SGS
  • Three factor Beam-Warming block tridiagonal with either central difference or Steger-Warming flux linearizations
  • Three factor scalar pentadiagonal (aka, Pulliam-Chaussee or DADI)
    • Three factor Diagonalized Diagonally Dominant ADI (D3ADI)
    • Unfactored SSOR
• The factored schemes have been the work horse methods of OVERFLOW, but they have some drawbacks
Background

• Four problems
  • Linearizations are mismatched
  • Boundary conditions are treated explicitly
  • It is difficult to develop intermediate BCs for the factored schemes
  • Factored schemes are already subject to factorization error
Background

- Four problems
  - Linearizations are mismatched
  - Boundary conditions are treated explicitly
  - It is difficult to develop intermediate BCs for the factored schemes
  - Factored schemes are already subject to factorization error

ADI Factorization (block tridiagonal matrix system):

\[
\begin{align*}
\begin{bmatrix}
I + \frac{\Delta t}{1+\theta} \partial_\xi A & I + \frac{\Delta t}{1+\theta} \partial_\eta B & I + \frac{\Delta t}{1+\theta} \delta_\xi C \\
\frac{\theta}{1+\theta} \Delta q^n - \frac{\Delta t}{1+\theta} RHS^n & + Error
\end{bmatrix} &= \Delta q^{n+1}
\end{align*}
\]

Factorization Error:

\[
Error = \left[ \frac{\Delta t}{1+\theta} \right] \left( \delta_\xi A \delta_\eta B + \delta_\xi A \delta_\xi C + \delta_\eta B \delta_\xi C \right) + \left[ \frac{\Delta t}{1+\theta} \right]^3 \left( \delta_\xi A \delta_\eta B \delta_\xi C \right) \Delta q^{n+1}
\]
Background

• Four problems
  • Linearizations are mismatched
  • Boundary conditions are treated explicitly
  • It is difficult to develop intermediate BCs for the factored schemes
  • Factored schemes are already subject to factorization error

ADI Factorization (block tridiagonal matrix system):

\[
\Delta q^{n+1} = \frac{\theta}{1+\theta} \Delta q^n - \frac{\Delta t}{1+\theta} RHS^n + \text{Error}
\]

Factorization Error:

\[
\text{Error} = \left( \frac{\Delta t}{1+\theta} \right)^2 \left( \frac{\Delta t}{1+\theta} \right)^3 \left( \frac{\Delta t}{1+\theta} \right)^3 \left( \frac{\Delta t}{1+\theta} \right)^3
\]

• One solution: implement fixes in the unfactored SSOR path
What fixes?

- Implicit physical BCs
- Improved linearizations
- CFL ramping
Implicit Physical BCs

• OVERFLOW’s BCs fall into three main types
  • $Q_{BC} = f(Q_{\text{Interior}})$
  • $Q_{BC} = f(Q_{\text{Interior}}, Q_{\text{Dirichlet}})$
  • $Q_{BC} = f(Q_{\text{Interior}}, Q_{BC})$

• The first type is fairly simple to linearize since they’re mainly extrapolations

• The second type is also simple, but require knowledge of which info is actually Dirichlet

• The third type requires a lot of math or help from a symbolic toolbox
What do we gain?

- Physical boundaries are now more closely coupled to the interior, resulting in faster propagation of information.
- For time accurate simulations, forces and moments converge faster within the subiteration process.

Airfoil Iterative Convergence
What do we gain?

- Physical boundaries are now more closely coupled to the interior, resulting in faster propagation of information.
- For time accurate simulations, forces and moments converge faster within the subiteration process.

Airfoil Iterative Convergence
Improved Linearizations

• In the current SSOR scheme, Steger-Warming FVS finite
difference Jacobians are used for the upwind schemes
  • Mismatch with the finite volume schemes
• In general, the better you do linearizing a problem, the
  better your nonlinear problem will converge (we hope)
• Roe, HLLC, HLLE++ fluxes have all been linearized
  • But not the low-Mach preconditioned versions (yet!)
What do we gain?

Airfoil Iterative Convergence

![Graph showing Airfoil Iterative Convergence with L2 Norm of Residual on the y-axis and Iteration Number on the x-axis. The graph compares Old Scheme and Old Scheme + Implicit BC’s.](image)
What do we gain?

- Much deeper iterative convergence, at the cost of \( \approx 15\% \) more time per iteration

**Airfoil Iterative Convergence**

![Graph showing iterative convergence](image)

- L2 Norm of Residual
- Iteration Number
CFL Ramping

• Ideally, we want to recover Newton’s method and let the time step become large

• Running with ITIME = 0 and DTPHYS = #, will force all cells to evolve at the same rate, i.e., constant time step and locally varying CFL number
  • ITIME of 1 or 2 does something similar

• With ITIME = 3, cells evolve at different rates, i.e., constant CFL and varying time step, so cells in the freestream are evolving faster than cells near the wall
  • ITIME of 4 does something similar
Switched Evolution Relaxation (SER)

• Developed by Mulder and van Leer*, it is probably the simplest CFL evolution scheme that ties CFL ramping to the nonlinear residual

\[ CFL^{n+1} = \varepsilon \times CFL^n \]

\[ \varepsilon = \frac{||R^{n-1}||}{||R^n||} \]

• The current implementation works with ITIME = 3 and allows for setting a starting CFLMAX on a grid by grid basis, but forms a global ramping factor
  • There are also CFL max and min limits

What do we gain?

Residual Convergence

L2 Norm of Residual vs. Step number
What do we gain?

Lift Convergence

- Central, Diagonal, ITIME=1
- Roe, SSOR, ITIME=1
- Roe, SSOR implicit, ITIME=1
- Roe, SSOR implicit, const CFL
- Roe, SSOR implicit, CFL ramping

Wall Time (sec) vs Lift Coefficient
Overall Speedups

• Depends on what your convergence criteria is and where you started to see convergence stall before

• In general, the more convergence you require, the more benefit you’re going to see
  • Time accurate, with 2 order drop, maybe 2x faster
  • Time accurate, with 3+ order drop, maybe 5x to 20x faster
  • Steady state, fully converged, maybe up to 100x faster
Examples

• S809 airfoil
• CFD 96 three element airfoil
• DPW-VI wing/body
• Generic quadcopter rotor
S809

• Transition model test case
  • Mach = 0.1, AoA = 1.0°, Re = 6 million
• 3rd-order Roe flux, fully turbulent SA-neg
  • Single block grid, 40k nodes, FMG and MULTIG
• Old scheme, max CFL ≈ 15
  • Can run higher, but the residuals will stall
  • Equivalent to a DTPHYS < 0.01
• New scheme, max CFL >10,000,000 with ramping
• In this case, just by switching from the old scheme to the new scheme, you gain a factor of 2 in number of iterations
• By using CFL ramping, you gain a factor of 100
CFD 96 Airfoil Validation

• AGARD-AR-303 test case A2-T2
  • Mach = 0.197, AoA = 20.18°, Re = 3.52 million
• 3\textsuperscript{rd}-order MUSCL, Roe flux, fully turbulent SA-neg
  • Overset, 85k node grid, FMG
  • Used scalar pentadiagonal, original SSOR, and new SSOR
CFD 96 Airfoil Validation

Lift Convergence

- **Scalar Pentadiagonal, ITIME=1, DT=0.1**
- **Original SSOR, ITIME=3, CFLMAX=20**
- **New SSOR, ITIME=3, CFLMAX_LIMIT=500**
CFD 96 Airfoil Validation

Pitching Moment Convergence

- Blue line: Scalar Pentadiagonal, ITIME=1, DT=0.1
- Black line: Original SSOR, ITIME=3, CFLMAX=20
- Green line: New SSOR, ITIME=3, CFLMAX_LIMIT=500

Cm

Wallclock time (sec)
DPW-VI

- Case 2A, 2.75 degree aerodynamically deformed geometry with alpha seek to find $C_L = 0.5$
  - Mach = 0.85, AoA ≈ 2.475°, Re= 5 million
- 5th-order WENO, HLLC flux, fully turbulent SA-neg
  - 14 million node ‘coarse’ grid, FMG and MULTIG
DPW-VI Residual Convergence

Collar grid begins to go unsteady due to side-of-body separation
DPW-VI Lift Convergence
Generic Quadcopter Rotor

- Two-bladed rotor in hover with time steps based on a 1/4 degree rotation

<table>
<thead>
<tr>
<th>Residual Convergence, per Rev</th>
<th>Time for 2 Order Drop, Hours</th>
<th>Time for 3 Order Drop, Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old SSOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New SSOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Speedup</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Generic Quadcopter Rotor

- Two-bladed rotor in hover with time steps based on a 1/4 degree rotation

<table>
<thead>
<tr>
<th>Residual Convergence, per Rev</th>
<th>Time for 2 Order Drop, Hours</th>
<th>Time for 3 Order Drop, Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old SSOR</td>
<td>14</td>
<td>&gt; 36</td>
</tr>
<tr>
<td>New SSOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Speedup</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Generic Quadcopter Rotor

- Two-bladed rotor in hover with time steps based on a 1/4 degree rotation

<table>
<thead>
<tr>
<th>Residual Convergence, per Rev</th>
<th>Time for 2 Order Drop, Hours</th>
<th>Time for 3 Order Drop, Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old SSOR</td>
<td>14</td>
<td>&gt; 36</td>
</tr>
<tr>
<td>New SSOR</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>Speedup</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Generic Quadcopter Rotor

- Two-bladed rotor in hover with time steps based on a 1/4 degree rotation
  - Speedup from a week to a weekend

<table>
<thead>
<tr>
<th>Residual Convergence, per Rev</th>
<th>Time for 2 Order Drop, Hours</th>
<th>Time for 3 Order Drop, Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old SSOR</td>
<td>14</td>
<td>&gt; 36</td>
</tr>
<tr>
<td>New SSOR</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>Speedup</td>
<td>2.8x</td>
<td>&gt; 4x</td>
</tr>
</tbody>
</table>
Generic Quadcopter Rotor

• Two-bladed rotor in hover with time steps based on a 1/4 degree rotation
  • Speedup from a week to a weekend
  • And faster time to a better converged solution, too!

<table>
<thead>
<tr>
<th>Residual Convergence, per Rev</th>
<th>Time for 2 Order Drop, Hours</th>
<th>Time for 3 Order Drop, Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old SSOR</td>
<td>14</td>
<td>&gt; 36</td>
</tr>
<tr>
<td>New SSOR</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>Speedup</td>
<td>2.8x</td>
<td>&gt; 4x</td>
</tr>
</tbody>
</table>
Still More To Do

• Reorder work/storage so Jacobians are calculated along with the fluxes
  • Currently, two passes are made through the residual/BC routines
• First-order linearizations work pretty well, but it would be nice to have the option to use exact Jacobians
  • Be able to handle nonlocal BC contributions
  • Be able to improve central scheme convergence
• Improve turbulence model linearizations
  • C-grid wake cuts can really hold up convergence of the turbulence models
• Provide localized flow field initializations
  • Nozzles converge better if they start subsonic rather than being initialized to supersonic values
Questions?